COLLEGE OF COMPUTING AND INFORMATION SCIENCES

Final Assessment of Lab Exam (Fall 2020 Semester)

Class Id	105032	**Course Title**	Numerical Computing
Program	BSCS	**Campus / Shift**	Main Campus, Morning
Date	23-11-2020	**Total Marks**	20
Duration	03 hours	**Faculty Name**	Muhammad Waqas Malik
Student Id	7791	**Student Name**	Muhammad Umair
Code			

Instructions:

- Fill out your Student ID and Student Name in above header.
- Do not remove or change any part question paper.
- Write down your answers with title “Answer for Question# 00”.
- Handwritten text or image should be on A4 size page with clear visibility of contents.
- In case of CHEATING, COPIED material or any unfair means would result in negative marking or ZERO.
- Each question carry equal marks.
- You need to print your name and id at the start of each code using print statement.
- <u>**Caution:**</u> Duration to perform Final Assessment is **02 hours only and 01 hour** is given to cater all kinds of odds in s **failed to upload answer sheet on LMS (in PDF format) within 3 hours limit, you would be considered as ABSENT**

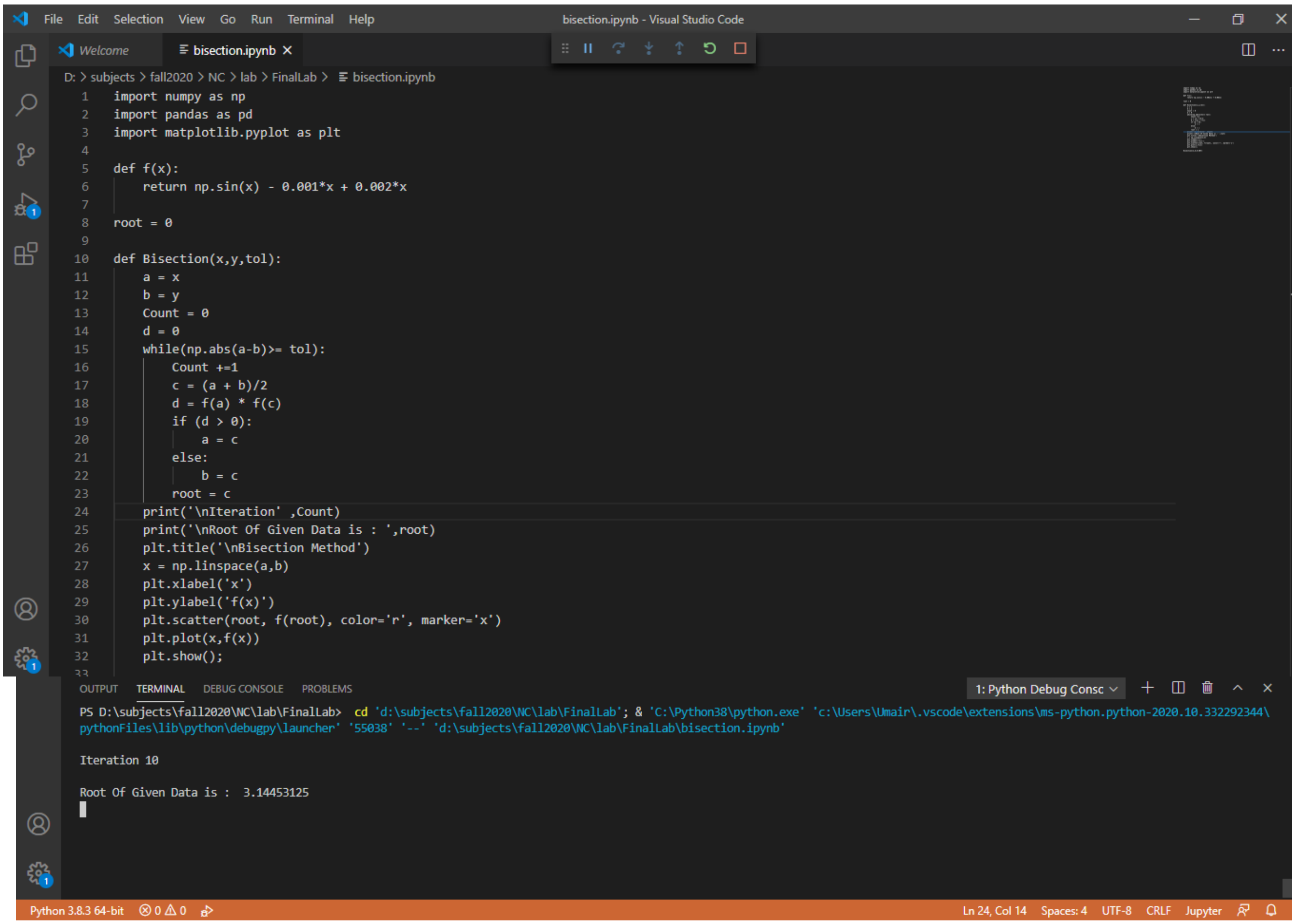
Fill the table with appropriate values			
Enter your Student ID in brackets		**R = [7791]**	
R[0]	**R[2]**	**R[-1]**	**R[-2]**
__7_	**__9_**	**__1_**	**_9__**

Question # 1:

Apply **Bisection Method** to given data and find the root.

a = R[-1] + 1
b = R[-1] + 5
tol = 0.00R[0]
f(x)=sin(x)−0.001x+0.002x

a=1+1 =>A=2
b=1+5 => B=6
tol = 0.007



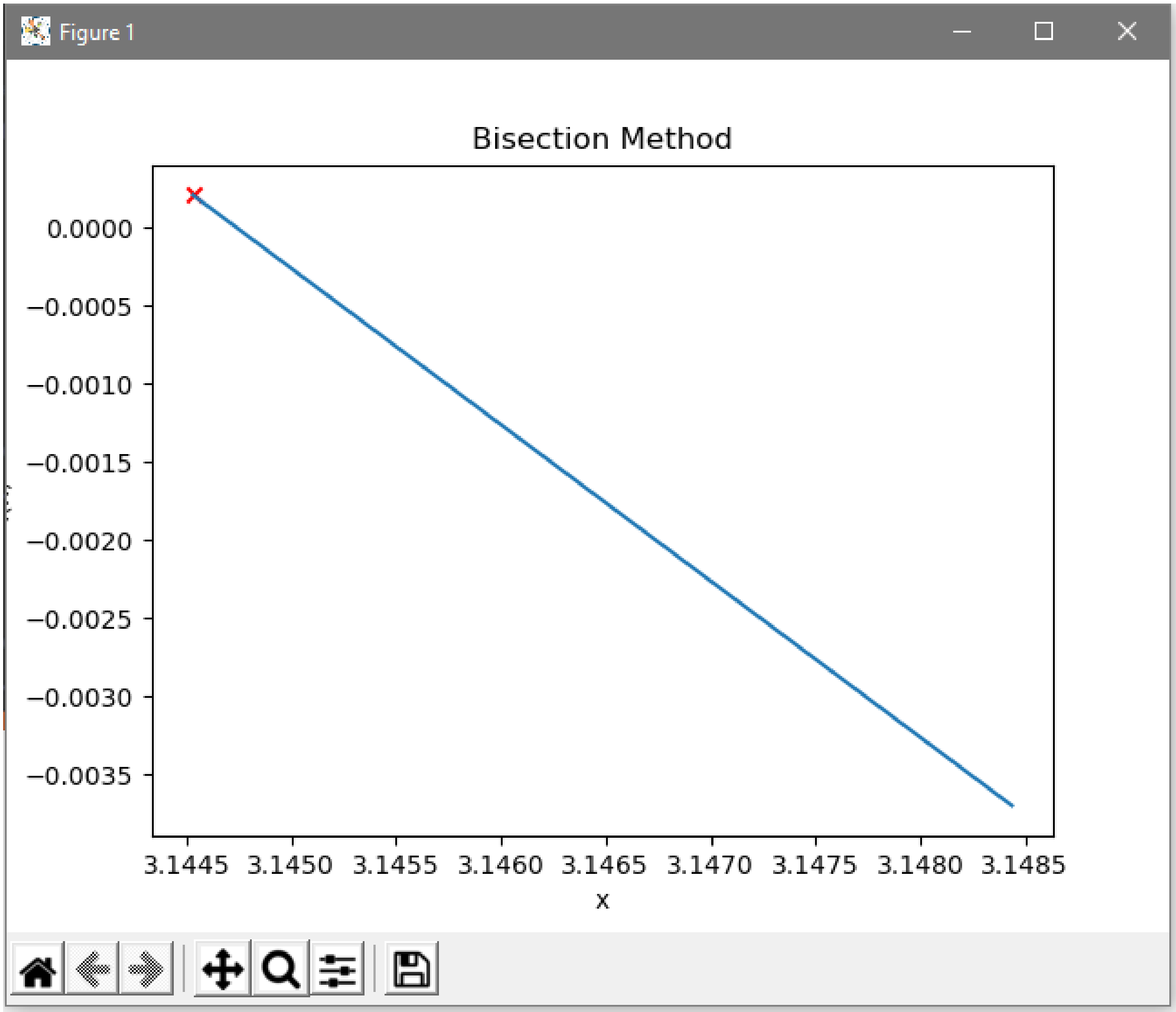
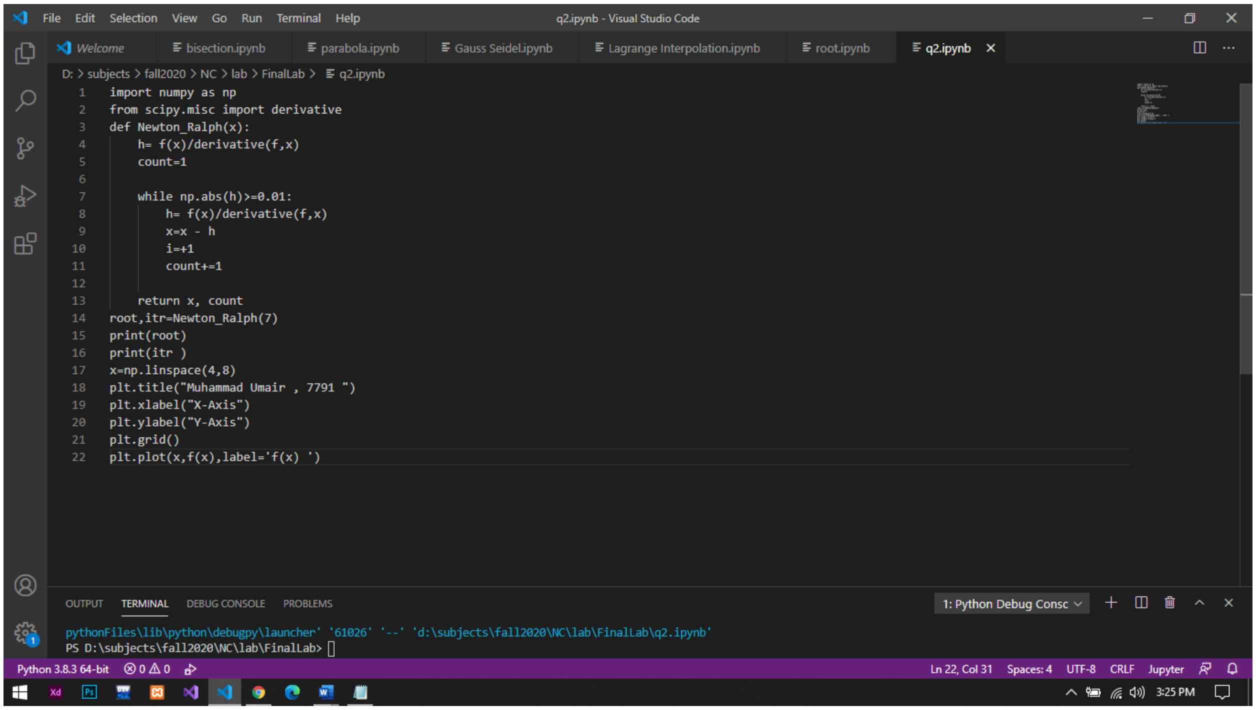


Figure 1
Bisection Method
0.0000
−0.0005
−0.0010
−0.0015
−0.0020
−0.0025
−0.0030
−0.0035
3.1445
3.1450
3.1455
3.1460
3.1465
3.1470
3.1475
3.1480
3.1485
x

Question # 2:

Assume you have a parabola, apply **convergence** on it for R[0] + R[-1] iterations. Your initial guess is R[-1] + 5 also plot the graph using matplotlib.



Question # 3:

Find the **root** however 1st step is given below.

$$x_{i+1} = x_i - \frac{x_i^2 - 2}{2x_i}$$

Initial Guess = R[-1] + 2

```python
from scipy.misc import derivative
import numpy as np
import matplotlib.pyplot as plt

def f(x):
    return (x**2 - 2)/(2*x)

al = list()
bl = list()

def NewtonRaphson(x):
    h = f(x)/ derivative(f, x)
    count = 1

    while np.abs(h) > 0.01:
        h = f(x)/ derivative(f, x)
        al.append(x)
        bl.append(h)
        x = x-h
        count += 1
    return count, x

IG = 1 + 2

itr, root = NewtonRaphson(IG)
print('iterations : ', itr)
print('root : ', root)


plt.title('Newton Raphson Method')
x = np.linspace(-3, 3)
plt.grid()
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.plot(x, f(x), label='f(x)')
plt.plot(x, 0*f(x), label='x')
plt.legend()
plt.scatter(root , f(root), color='blue')
```

```
PS D:\subjects\fall2020\NC\lab\FinalLab>  cd 'd:\subjects\fall2020\NC\lab\FinalLab'; & 'C:\Python38\python.exe' 'c:\Users\Umair\.vscode\extensions\ms-python.python-2020.10
pythonFiles\lib\python\debugpy\launcher' '60822' '--' 'd:\subjects\fall2020\NC\lab\FinalLab\root.ipynb'
iterations :  8
root :  1.411528212528584
PS D:\subjects\fall2020\NC\lab\FinalLab>  cd 'd:\subjects\fall2020\NC\lab\FinalLab'; & 'C:\Python38\python.exe' 'c:\Users\Umair\.vscode\extensions\ms-python.python-2020.10
pythonFiles\lib\python\debugpy\launcher' '60832' '--' 'd:\subjects\fall2020\NC\lab\FinalLab\root.ipynb'
iterations :  8
root :  1.411528212528584
PS D:\subjects\fall2020\NC\lab\FinalLab>
```

Question # 4:

Use the **Gauss Seidel** Method with relaxation to solve Ax=b, where

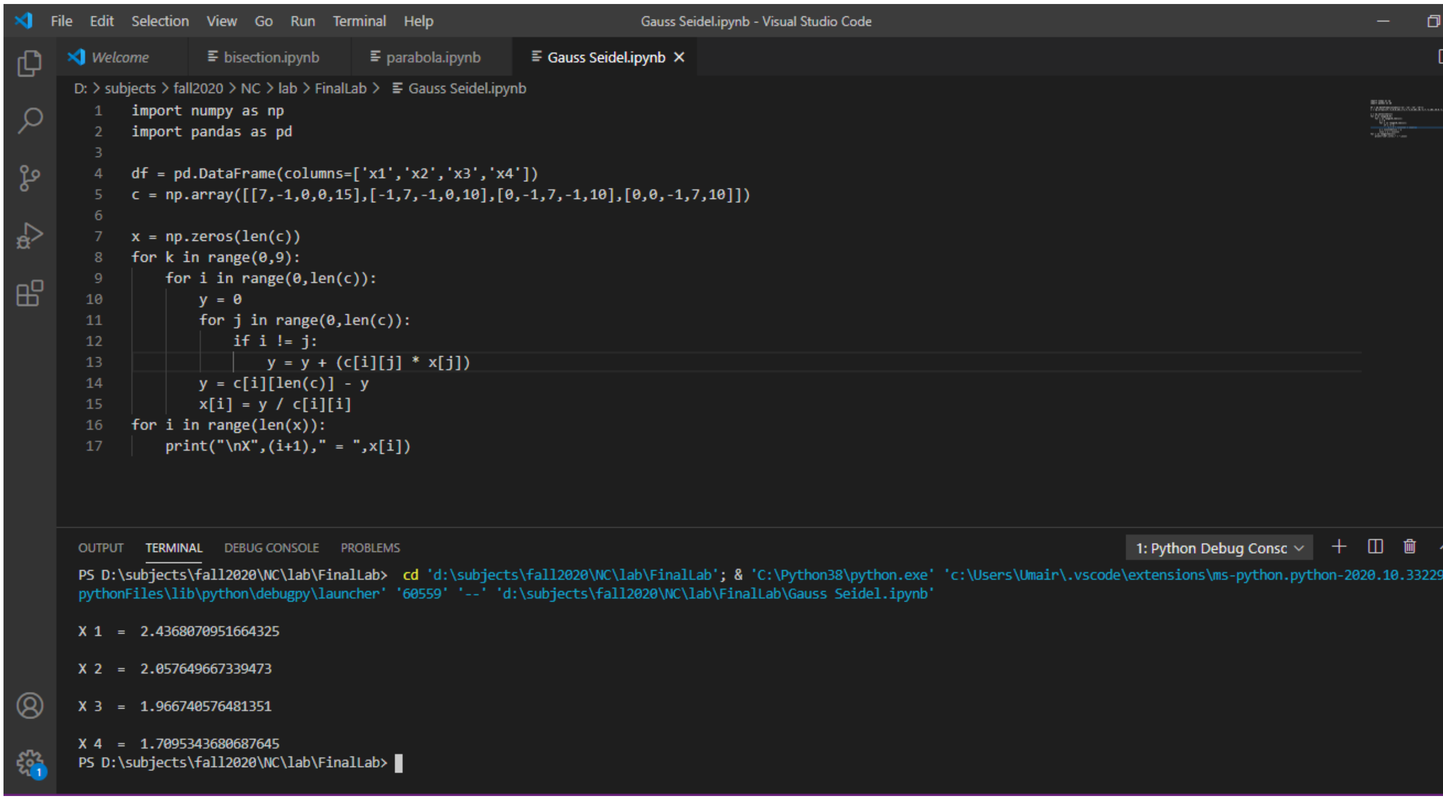
$$\begin{bmatrix} R[0] & -1 & 0 & 0 \\ -1 & R[0] & -1 & 0 \\ 0 & -1 & R[0] & -1 \\ 0 & 0 & -1 & R[0] \end{bmatrix}, \text{ b } = \begin{bmatrix} 15 \\ 10 \\ 10 \\ 10 \end{bmatrix},$$

M. UMAIR , 7791

QNo4:

$$\begin{bmatrix} R[0] & -1 & 0 & 0 \\ -1 & R[0] & -1 & 0 \\ 0 & -1 & R[0] & -1 \\ 0 & 0 & -1 & R[0] \end{bmatrix} \begin{bmatrix} 15 \\ 10 \\ 10 \\ 10 \end{bmatrix}$$

$$\begin{bmatrix} 7 & -1 & 0 & 0 \\ -1 & 7 & -1 & 0 \\ 0 & -1 & 7 & -1 \\ 0 & 0 & -1 & 7 \end{bmatrix} \begin{bmatrix} 15 \\ 10 \\ 10 \\ 10 \end{bmatrix}$$



Question # 5:

First fill the given table with appropriate values and apply **Lagrange Interpolation** for the given data.

x	**R[-1] -5 =**	**R[-1] -2 =**	**R[-1] =**	**R[-1] + 2 =**	**R[-1] + 5 =**	**R[-1] + 7 =**
Y	**2 * x - 1= ___**	**2 * x - 0= ___**	**2 * x – 1 = ___**	**2 * x – 0 = ___**	**2 * x - 1= ___**	**2 * x - 1= ___**

Find the value of

i) **xp = R[-1] + 4** **ii) xp = R[-1] - 3**

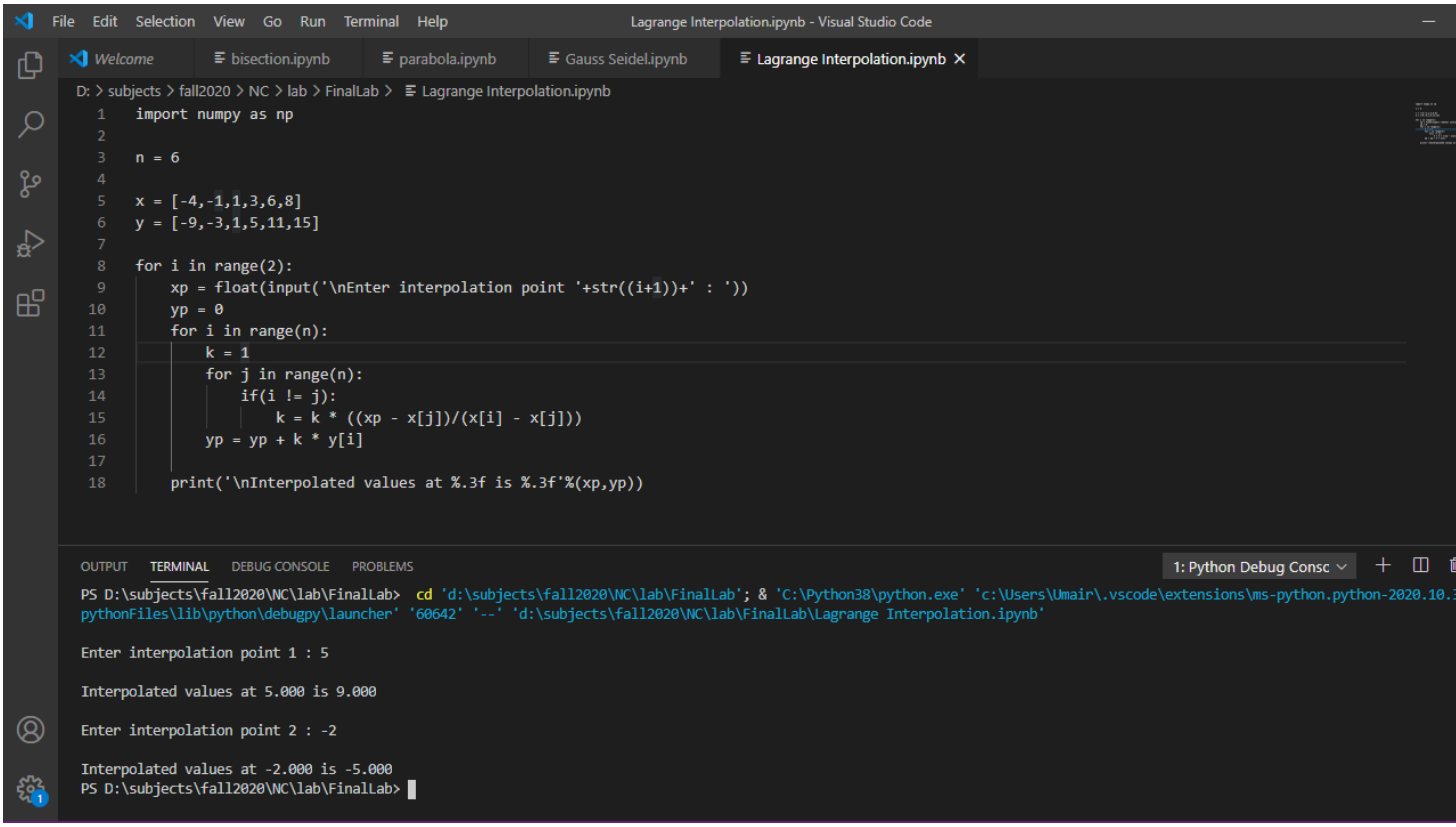
Q No 5: S-Id: 7791

$R[-1] = 1$

X	Y
$1-5 = -4 = x_0$	$2\times(-4)-1 = -9$
$1-2 = -1 = x_1$	$2\times(-1)-1 = -3$
$1 = 1 = x_2$	$2\times 1-1 = 1$
$1+2 = 3 = x_3$	$2\times 3-1 = 5$
$1+5 = 6 = x_4$	$2\times 6-1 = 11$
$1+7 = 8 = x_5$	$2\times 8-1 = 15$

i) $xp = 1+4 = 5$

ii) $xp = 1-3 = -2$


File Edit Selection View Go Run Terminal Help
Lagrange Interpolation.ipynb - Visual Studio Code
Welcome
bisection.ipynb
parabola.ipynb
Gauss Seidel.ipynb
Lagrange Interpolation.ipynb
D: > subjects > fall2020 > NC > lab > FinalLab > Lagrange Interpolation.ipynb
import numpy as np
n = 6
x = [-4,-1,1,3,6,8]
y = [-9,-3,1,5,11,15]
for i in range(2):
    xp = float(input('\nEnter interpolation point '+str((i+1))+' : '))
    yp = 0
    for i in range(n):
        k = 1
        for j in range(n):
            if(i != j):
                k = k * ((xp - x[j])/(x[i] - x[j]))
        yp = yp + k * y[i]
    print('\nInterpolated values at %.3f is %.3f'%(xp,yp))
OUTPUT
TERMINAL
DEBUG CONSOLE
PROBLEMS
1: Python Debug Consc
PS D:\subjects\fall2020\NC\lab\FinalLab> cd 'd:\subjects\fall2020\NC\lab\FinalLab'; & 'C:\Python38\python.exe' 'c:\Users\Umair\.vscode\extensions\ms-python.python-2020.10.
pythonFiles\lib\python\debugpy\launcher' '60642' '--' 'd:\subjects\fall2020\NC\lab\FinalLab\Lagrange Interpolation.ipynb'
Enter interpolation point 1 : 5
Interpolated values at 5.000 is 9.000
Enter interpolation point 2 : -2
Interpolated values at -2.000 is -5.000
PS D:\subjects\fall2020\NC\lab\FinalLab>